SPRING BOOT:

What is Spring boot: Spring Boot is a Java-based open-source framework used to create stand-alone, production-grade Spring Applications with minimal configuration.

It is an extension of the Spring Framework that eliminates boilerplate code and XML configuration. It comes with embedded servers like Tomcat/Jetty, auto-configuration, and production-ready features.

Key Features of Spring Boot:

1. Auto-Configuration - Automatically configures application based on dependencies.

2. Embedded Server - No need to deploy WAR files, comes with embedded Tomcat.

3. Stand-Alone - Can run application using java -jar command.

4. Production-Ready - Provides Actuator, Health Checks, and Metrics.

In short, Spring Boot helps developers to create microservices and web applications quickly and easily.

SPRING BOOT FEATURES

Spring Boot Features:

1. Auto-Configuration:

Automatically configures Spring application based on jar dependencies. No need for manual XML configuration.

2. Embedded Servers:

Comes with embedded Tomcat, Jetty and Undertow. No need to deploy WAR file.

3. Stand-Alone Applications:

Creates stand-alone applications that can be run using java -jar.

4. Starter Dependencies:

Provides starters like spring-boot-starter-web, spring-boot-starter-data-jpa to simplify Maven dependencies.

5. Spring Boot Actuator:

Provides production-ready features like health check, metrics, auditing.

6. No XML Configuration:

No need for XML, uses annotation based configuration.

7. Spring Initializr:

Web tool to quickly generate Spring Boot project structure.

8. Opinionated Defaults:

Provides default configurations but allows to override if needed.

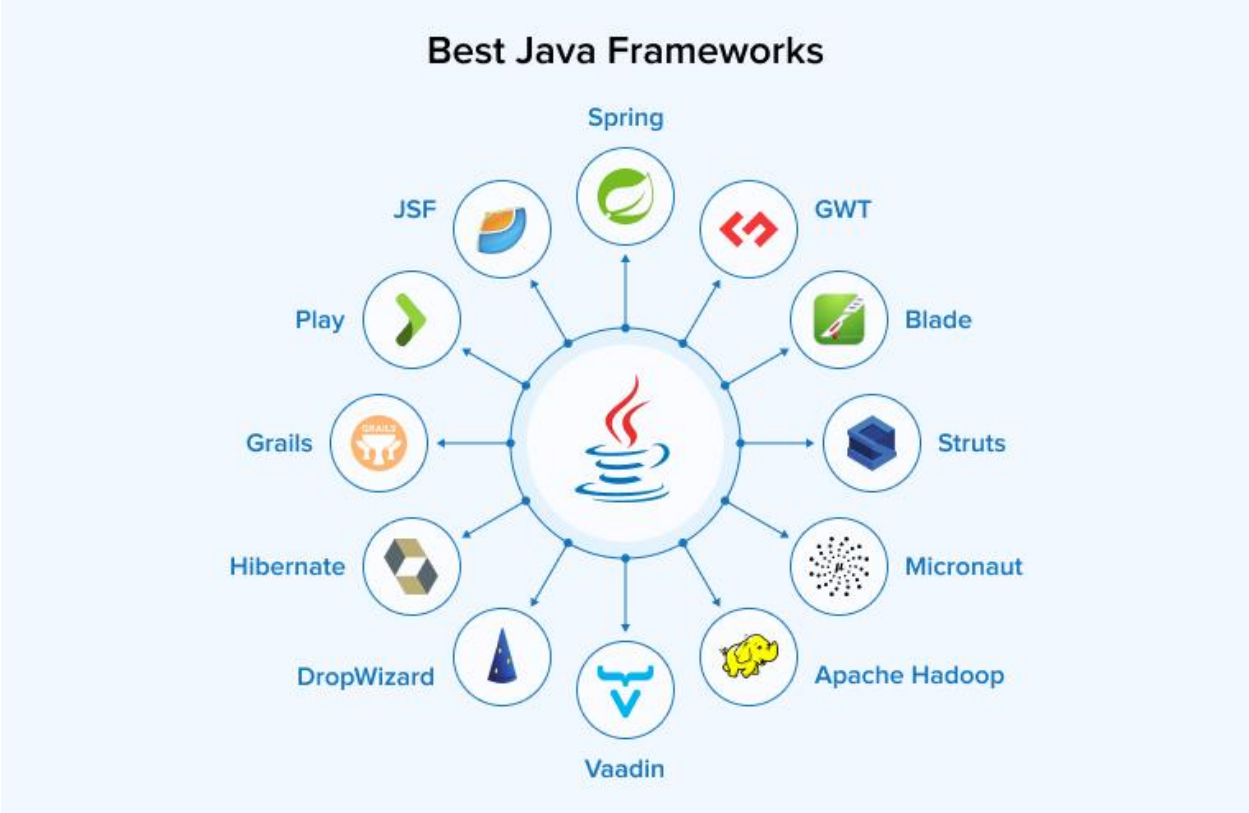
9. Microservices Support:

Best framework to create REST APIs and Microservices quickly.

10. DevTools:

Provides auto-restart and live reload for faster development.

JAVA FRAMEWORK



Java Framework is a collection of pre-written code, tools, libraries and best practices that helps developers to build Java applications quickly and efficiently without writing code from scratch.

In simple words, framework is a ready-made structure for building applications. It provides a foundation so that developers can focus on business logic instead of basic functionality.

What is Java Framework?

A Java framework is a platform for developing software applications. It provides a way to write code in a structured and reusable manner. Framework follows Inversion of Control (IoC) principle where framework controls the flow of the program.

Types of Java Frameworks:

1. Web Frameworks:

Used to create web applications and websites.

Examples: Spring, Spring Boot, Struts, JSF (JavaServer Faces), Vaadin, Play.

2. ORM Frameworks:

Used to connect Java application with database without writing complex SQL.

Examples: Hibernate, EclipseLink, Java Persistence API (JPA).

3. Testing Frameworks:

Used for unit testing and integration testing.

Examples: JUnit, TestNG, Mockito.

4. Microservices Frameworks:

Used to build microservices architecture based applications.

Examples: Spring Boot, Spring Cloud, Micronaut, Quarkus.

5. Build Frameworks:

Used to manage dependencies and build process.

Examples: Maven, Gradle.

Most Popular Java Framework - Spring Framework:

Spring is the most popular and powerful Java framework. It provides comprehensive infrastructure support for developing Java applications. Spring is based on two core concepts: Dependency Injection (DI) and Aspect Oriented Programming (AOP).

Spring Boot is an extension of Spring which makes it even easier to create production-ready applications with minimal configuration. Today 90% of Java jobs require Spring Boot.

Other Popular Frameworks:

Hibernate: Hibernate is an ORM framework that maps Java objects to database tables. It eliminates JDBC boilerplate code and provides HQL for database operations.

Struts: It is an open-source MVC framework for creating enterprise-level web applications. It is old but still used in many legacy projects.

JSF: JavaServer Faces is a component-based framework from Oracle for building user interfaces.

Why Use Java Framework?

1. Faster Development: No need to write common code again and again.
2. Code Reusability: Code can be reused across projects.
3. Security: Frameworks provide built-in security features.
4. Scalability: Applications built with frameworks are easy to scale.
5. Community Support: Popular frameworks have large community support.
6. Maintainability: Code is structured and easy to maintain.
7. Testing Support: Easy to test applications.

Advantages of Java Framework:

- Reduces development time and cost
- Provides best coding practices
- Built-in libraries and tools
- Automatic management of dependencies
- Better performance and security

Disadvantages:

- Learning curve for beginners

- Framework overhead can affect performance sometimes

- Dependency on framework updates

Conclusion:

Java frameworks are essential for modern Java development. Without frameworks, building enterprise-level applications would be very difficult and time-consuming. For beginners, it is recommended to start with Core Java, then learn Spring and Spring Boot because they are most demanded in the industry. A framework does not replace Java knowledge; it makes a Java developer more productive and professional. Choosing the right framework depends on project requirements, but Spring Boot is currently the industry standard for Java development.

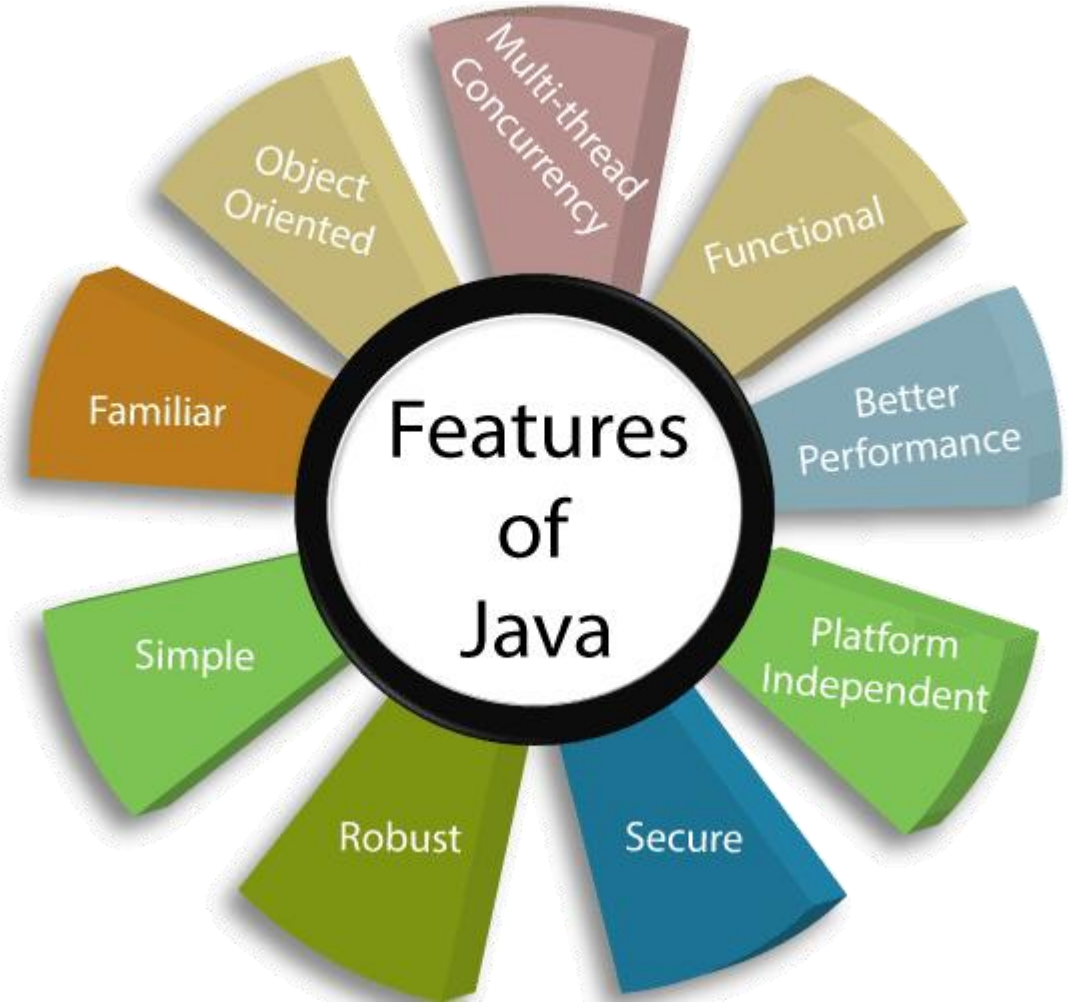
JAVA FEATURES

Java Features / Characteristics of Java

Java is one of the most popular and widely used programming languages in the world. It was developed by James Gosling at Sun Microsystems in 1995. Java is known for its unique features that make it powerful, secure and platform-independent. The main features of Java are as follows:

1. Simple:

Java is simple and easy to learn. Its syntax is clean and similar to C++. Java removed complex features like pointers, operator overloading and manual memory management which makes it easy to understand for beginners.



2. Object-Oriented:

Java is a pure object-oriented programming language. Everything in Java is an object. It follows OOP principles like Inheritance, Encapsulation, Polymorphism and Abstraction. This helps to create modular, reusable and organized code.

3. Platform Independent:

This is the best feature of Java. Java follows WORA principle - Write Once Run Anywhere. Java code is compiled into bytecode which can run on any platform that has JVM (Java Virtual Machine). Unlike C++ which is platform dependent, Java is platform independent.

4. Portable:

Java is portable because bytecode can be carried to any platform. Bytecode is platform independent and can run on any operating system like Windows, Linux, Mac.

5. Secured:

Java is highly secured. It provides security at many levels. It does not use pointers which can cause security issues. Java programs run inside JVM sandbox. It also provides bytecode verification, security manager and access control.

6. Robust:

Robust means strong. Java is robust because it has strong memory management, exception handling and automatic garbage collection. It checks errors at compile time and runtime. Garbage collector automatically removes unused objects.

7. Architecture Neutral:

Java compiler generates bytecode which is not related to any particular machine architecture. This bytecode can run on any machine, making Java architecture neutral.

8. Multithreaded:

Java supports multithreading which means multiple tasks can run at the same time in a single program. This helps to achieve maximum CPU utilization. For example, one thread can play music while another thread downloads a file. This feature is very useful for games and multimedia applications.

9. Distributed:

Java is distributed because it allows to create distributed applications using RMI (Remote Method Invocation) and EJB. We can access data and programs from different computers on a network.

- 10. High Performance:

Java performance is high because of Just-In-Time (JIT) compiler. JIT converts bytecode into native machine code at runtime which improves speed. Although Java is slower than C++ due to interpretation, its performance is much better than other interpreted languages.

11. Dynamic:

Java is dynamic because it can adapt to evolving environment. It supports dynamic loading of classes, dynamic compilation and automatic memory management.

12. Interpreted:

Java is both compiled and interpreted. Source code is compiled to bytecode and then interpreted by JVM. This two-step process makes Java secure and portable.

13. Automatic Garbage Collection:

Java automatically manages memory. It has a garbage collector that automatically deletes unused objects and frees memory. Programmer does not need to delete objects manually like in C++.

14. Rich API:

Java provides rich Application Programming Interface with thousands of built-in classes and methods for networking, database, I/O, utilities, etc.

15. Open Source:

Java is open source and free to use. Large community support is available.

16. Case Sensitive:

Java is a case-sensitive language. It differentiates between uppercase and lowercase letters.

17. Inheritance Support:

Java supports inheritance, which allows one class to acquire the properties of another class.

Due to these powerful features, Java is used everywhere - from mobile applications (Android), web applications, enterprise applications, desktop applications to scientific applications. Java's platform independence, security, object-oriented nature and robustness make it the first choice for developers and companies. Learning Java opens huge career opportunities because Java is the backbone of modern enterprise development. These features are the reason why Java has remained popular for more than 25 years and still dominates the software industry.

CONTACT US

EXCELLENCE TECHNOLOGY

+91 7710788822

info
@excellencetechnology.in

near Shiv Mandir, opposite the Cooperative Kangra Bank in the Main Bazar / Lower Bazar area of Hamirpur, Himachal Pradesh (PIN 177001).